Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 92 (2016) 487 – 492

Procedia Computer Science

2nd International Conference on Intelligent Computing, Communication & Convergence

(ICCC-2016)

Srikanta Patnaik, Editor in Chief

Conference Organized by Interscience Institute of Management and Technology

Bhubaneswar, Odisha, India

Ontology Based Natural Language Interface for Relational Databases

B.Sujatha*[a], Dr.S.Viswanadha Raju[b]

a. *Assistant Professor, Department of CSE, Osmania University, Hyderabad, India*

b. *Professor, Dept. Of CSE, JNTU college of Engineering, Jagityal, Karimnagar, India*

Abstract

Developing Natural Language Query Interface to Relational Databases has gained much interest in research community since forty years. This can be termed as structured free query interface as it allows the users to retrieve the data from the database without knowing the underlying schema. Structured free query interface should address majorly two problems. Querying the system with Natural Language Interfaces (NLIs) is comfortable for the naive users but it is difficult for the machine to understand. The other problem is that the users can query the system with different expressions to retrieve the same information. The different words used in the query can have same meaning and also the same word can have multiple meanings. Hence it is the responsibility of the NLI to understand the exact meaning of the word in the particular context. In this paper, a generic NLI Database system has proposed which contains various phases. The exact meaning of the word used in the query in particular context is obtained using ontology constructed for customer database. The proposed system is evaluated using customer database with precision, recall and f1-measure.

Peer-review under responsibility of the Organizing Committee of ICCC 2016

Keywords: Natural Language queries, Ontology, Relational Databases, First Order Logic, Structured Language Query, Backus Naur Form, N-grams.

*Corresponding Author. Tel: 08977207889

E-Mail Address: drpvijayapalreddy@gmail.com

1. Introduction:

Database systems are used since 1970s for the storing various kinds of data for different purposes such as commercial and personal needs. Though there are many types of architectures for database design like object oriented, object based, file based, hierarchical based and network based, the predominant designing of databases follow relational database architecture to store the data by using various types of storage devices. In relational databases, the data is stored using tables. The table contains set of rows and columns. Each column represent and attribute and each represents the instance of the data for a set of attributes. The data can be manipulated using various operators with fixed set of keywords by following a set syntax rules. By learning this structured query language one can extract the required data from the whole set of data, can also perform various operations such as update, manipulate and deletion of the data.

Peer-review under responsibility of the Organizing Committee of ICCC 2016

doi:10.1016/j.procs.2016.07.372

The Relational database management systems are more popular based on the characteristics like its robustness and flexibility, high performance, scalability, data security and protection and flexible data maintenance. Above all these advantages, it allows to index, perform aggregation, filtering and sorting can be done on the data using structured query language.

There are some disadvantages with relational databases. To perform operations on the data which is stored on databases, it is required to learn the structured query language. Hence , the naive user who knows only the natural language can not directly access the required information from the databases. To come out from these limitations, it is required to design a tool which can understand the requirements of the naive user through natural language query, convert the natural language query into an equivalent structured language query. Then the obtained structural query is used to access the required information from the databases. This kind of tool ins termed as Natural Language Interface to Databases or NLIDB system. Thus, the NLIDB system take the input as natural language query and converts it into a structures language query and returns the desired information to the naive user.

The designing of a NLIDB system for various languages and for different underlying databases is attempted by various researchers since five decades. But, designing of an most suitable NLIDB systems with high accuracy, precision and recall is still an open research problem which need to be addressed. The various earlier developed NLIDB systems focused on particular databases. There is need of designing a generic NLIDB system which can address the robustness and scalability of the applications. It is required to attempt the problem of portability to customize a NLIDB system to a other language and to other set of datasets designed for various domains. The efficiency of conventional NLIDB systems depend mostly on domain experts capabilities and linguistic features of the natural language.

In this paper, it is focused on designing a NLIDB system to overcome the various issues such as portability to different languages and to access the required information independent of the underlying database. It also required maintains the scalability and robustness of the system. The word sense disambiguation is achieved using N-grams and ontology which is constructed on the customer database. The remaining sections explains about the related work, proposed model, word sense disambiguious using N-gram and ontology, experimental evaluation and conclusions and possible future extensions.

2. Related Work:

There are many designing models are proposed in the literatures in the field of NLIDB such as pattern matching systems, syntax based systems, semantic based grammar systems and intermediate representation of languages system. The pattern matching systems takes input as a set of rules and sample set of patterns. Based on the inputted word of sentence with natural language, it will be compared with the predefined patterns [1]. If there is a match between the input and predefined pattern then an action will be generated and these generated actions will be stored in the database. The response given to the user is based on the action generated. This kind of systems are limited to specific databases. The accuracy of the system is depend on the complexity of the patterns used to train and based on the set of rules used to train the system [2]. The NLIDB system SANVY is a good example for pattern- matching systems [3].

The syntax based systems takes the user query as input and parse the given input syntactically. The parse tree generated for the input query is overlapped with the one structured query of the database expressed using structured query language. LUNAR is a best example for syntax based NLIDB systems [4]. In these systems, the grammar rules are derived to match the various user questions with syntactic structures [5]. This system is used to answers the questions on rocks which were collected from the moon. With the corrections in the dictionary errors, the performance of the system has increased [8].

In the semantic grammar system, the parse is simplified by eliminating unimportant nodes or by combining two or more nodes into one node. The complexity of structured query can be reduced in semantic grammar system. Semantic grammar systems are more simpler when compared with syntax based systems. But these systems need to be trained with a prior knowledge of the various elements of a domain. PLANES and LADDER are the good examples for Semantic grammars systems [6,7].

In many NLIDB systems, the natural language query is transformed into an intermediate logical query. The logical query is represented using a meaningful representative language such as first logic language or Boyce Codd normal form. This kind of representative languages, represents the meaning of the users queries in high order level of concepts. These concepts are independent from the structure of the database. This representative query is then transformed into an expression in the structured query language which can extract the relevant data from the databases.

In the intermediate representation of natural language systems, the natural language query is inputted to the system. This query is processed for syntax rules using a parser. Based on the set of syntax rules of a natural language, it generates a parse tree. By using the semantic rules of semantic interpreter module, the generated parse tree is translated into an intermediate logic query. In the semantics rule, left hand side of the syntax rule contains the logic expression of the constituent where as right-hand side of the syntax rule is a function of the logic expressions of the constituents. The logic expressions represents the words which are corresponds to lexicon. To get the required information from the database, the logic query is to be transformed into a structured query which is supported by the underlying Database Management System. MASQUE/SQL is an example of intermediate

representation language systems [7].

By using semantic grammar techniques which interleaves semantic and syntactic processing in distributed databases, LADDER system is used to parse natural language questions to database understandable queries [7]. The another NLIDB system implemented using the language called Prolog was CHAT-80. This system transforms the natural language inputted English queries into Prolog expressions. These Prolog expressions are evaluated using the Prolog database. ROBOT which was a prototype of a NLIDB system named INTELLECT which was a commercial natural language interface to database systems [9]. ASK is the another NLIDB system which allows the users to train the system with new words and concepts while interacting with the system. By using the system, it is possible to make interactions with various external sources such as external databases, chating, Facebook, twitter, email programs and many other applications.

Generic Interactive Natural Language Interface to Databases (GINLIDB) was designed by the using UML and developed using Visual Basic.NET. The system was a generic system and it works for underlying suitable database and knowledge base [10]. SynTactic Analysis using Reversible Transformations (START) is also another Natural Language System. It was the first Web-based question answering system. It was available online and continuously operating till now [11]. It utilizes various language Dependant functions such as parsing, semantic analysis, word sense disambiguous, natural language annotation for appropriate information segmentation and presentation for the user [12].

JUPITER was a NLIDB system to know the weather information worldwide. The user can pose a question to the system in their native language to forecast the weather information over the telephone. The Oracle Structured Query Language SQL can be learned by the students using the NLIDB system called SQL-Tutor. If the student asked the new questions by typing at terminal then also, the SQL-Tutor can answer the question by using the existing knowledge [13]. KUQA system divides the query based on possible answer and after that it uses NLP techniques and also WorldNet to identify the answers which suitable to its corresponding category. But, this system can not handle any linguistic information [11]. QuALiM another NLIDB system designed based on complex syntactic structure which were based on certain syntactic description question patterns [11].

3. The NLIDB model:

The proposed model contains takes natural language query is input from the naive user and then the inputted query is translated into Structured Query using various phases. The various phases are as follows:

1) Stop word Removal
2) Stemming
3) Content word extraction
4) Syntactic analysis
5) Semantic analysis
6) Candidate Query formulation

Initially the system is inputted with the natural language query through the natural language interface. From the NL query the set of stopwords are eliminated using predefined set of stop word list. In the second phase, the remaining words in the query are processed for root word extraction. The set of root words which are resulted from the second phase are considered as meaningful words. These words were assigned with parts of speech tagging using natural language toolkit. In the syntactic parsing phase, the query is parsed using top - down parser. Parsing of the given query is done using First Order Logic which is used mainly in the field of artificial intelligence. Backus Norm Form is used to represent the first order logic of the inputted query.

The semantic analysis is performed using ontology, N-grams. The ambiguity in the meaning of the word is resolved using N-gram technique and ontology which is constructed on the customer database.

3.1 N-grams:

The meaning of a word can be correctly understood when words are taken with their neighbours. The language model which exploits on the order of words is called n-gram language models, where n is non-zero positive integer. N-gram model can be assumed as a small movable window sliding over text of size ‘n’. Based on the size of the window the model can be termed as unigram, bigram, trigram and multigram. The semantic meanings of words can be diagnosed using n-grams from a sentence. Word when taken in consideration alone can lead to ambiguity rather that taken in context, this is achieved using n-grams of variable size depending on the level of ambiguity. If ambiguity persists then the n-grams size is increased to obtain the accurate meaning.

3.2 Ontology:

Building ontology for CPVbase involves analysis of every word that can be associated with the database. The words and the corresponding actions to be taken to convert the word into formal language are specified in this ontology. Thus the ontology is

specific with the database used, that is CPVbase. When a word is encountered it has to assigned senses in the corpus. It is a known fact that words in a language are divided into two categories namely content word and function word. Content words are verb, noun, adjective and adverb. And the function words are preposition, conjunction, pronoun and interjection. Every language definitely has repositories of content words and function words.

The main information load is carried by the content words and among content words internally nouns carry the maximum amount of information followed by verbs, adjectives and adverbs comes as qualifiers for nouns and verbs respectively. Function words are close category words that define a language. Addition or deletion of function words is rarely seen where as content words are added frequently due to exigency or technological developments. he ontology that is created for the CPVbase has attempted to have POS marked corpus and Sense marked corpus. Every word has different synonyms, the collection of distinct meanings associated with a word is called as Synset, means its synonyms set. The information of Synset of each word if known then the word can be sensed correctly.

The ontology contains the information about the tables of the CPVbase, their contents such as table attributes and table fields. The vocabulary of natural language that can be used respective to the database is also stored with the relevant processing details. Each of the word stored in the ontology corresponds to a table and is associated with an SQL clause. When a user asks his question to find the information from CPVbase, the natural language statement after undergoing the stemming and stop words removal gives the keywords to be processed to generate the appropriate SQL query. The keywords are parsed and mapped according to the information from ontology that is related to CPVbase. Each word of the database is stored with their possible synsets so that the ambiguous words are sensed correctly.

In addition to the wordnet, the ontology also has details for phrases comprising prepositions, adjectives, conjunctions such as less than, greater than, equal to, negation, above, below etc that requires mathematical sort of processing to be done. The adverbials of time such as last month, previous month, this morning etc must also be taken care. Thus the ontology is the crust of the system, if the ontology is built by taking into consideration all possibilities of a user need's specification then the natural language interface to the database can said to be efficient. But the natural language processing system has always been a restrictive framework due to incomplete language coverage.

The natural language statement provided by the user need not contain the above vocabulary as it is, there are also possibilities of words addressed in different manner though the meaning is same. Such words are put to scrutiny to map them to the ontological words. The words are mapped with the terms contained in the ontology, the mapping is not one to one as in English language one word can be expressed in many forms therefore the mapping which can arise here could be one to many or many to many. Also when a user makes mistakes while specifying his need, the mistakes such as spelling errors, grammatical mistakes must be absorbed.

Candidate query formulation is performed using EFFECN algorithm. This algorithm deals with splitting of the natural query, joining of the tables and selecting multiple columns and multiple rows based on the conditions specified in the query.

4. The experimental evaluation:

The EFFCN system performance is measured in terms of retrieval efficacy using the information retrieval system metrics known as precision and recall. The attainment of relevant information by the user as per the natural language query in English gives the retrieval efficacy. The precision is the measure of retrieved results that are relevant to the need, evaluated using the fraction of relevant documents retrieved to the total number of documents retrieved.
Mathematically it can be expressed as

$$Precision = \frac{CorrectlyAnsweredQueries}{AnsweredQueries} \quad (4.1)$$

Recall measures the relevant results retrieved as per the user statement. That is the fraction of relevant documents retrieved to the total number of relevant documents present in the system. Based on the precision and recall measurements, the system was tested for a random of 100 queries.

$$Recall = \frac{CorrectlyAnsweredQueries}{TotalNumberofQueries} \quad (4.2)$$

The results shows that the system offers a recall rate of 0.84 which means that it has 84% Probability of generating correct responses to the user queries. This proves the effective and optimal working of the system. The Precision and recall graph for varying number of queries is presented in figure 4.1.

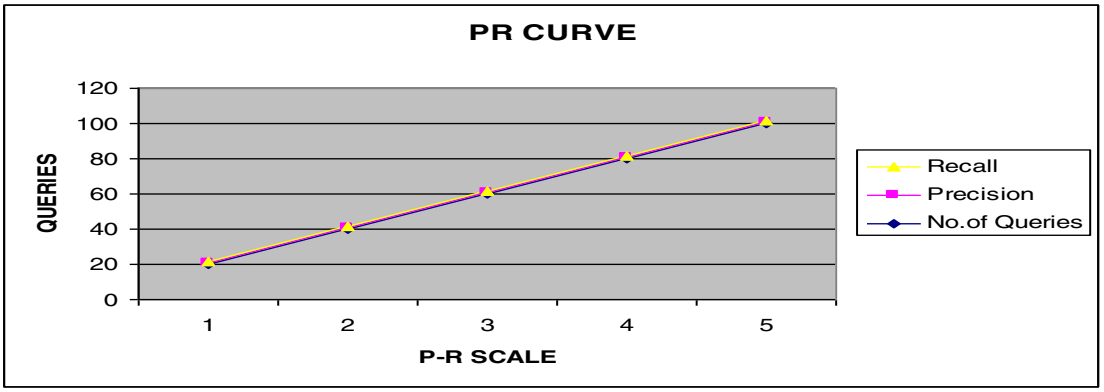

Figure 4.1: The Precision- Recall Curve for varying number of queries

The X-axis holds the Precision and Recall Values with respect to the number of queries. The Precision and Recall are constant showing the desirable and obtained relevancies are near approximately. Another graph plotted in the figure 4.2 shows the Precision and recall variations. The graph plotted in the figure 4.2 represents the relation between Precision and Recall with respect to the EFFCN system. The curvy edges in the graph change in the precision with the minor effect in the recall. The precision and recall is decreasing if irrelevant queries are observed. Thus the precision and recall can be well defined if the data search is acquired with maximum relevant terms in the query.

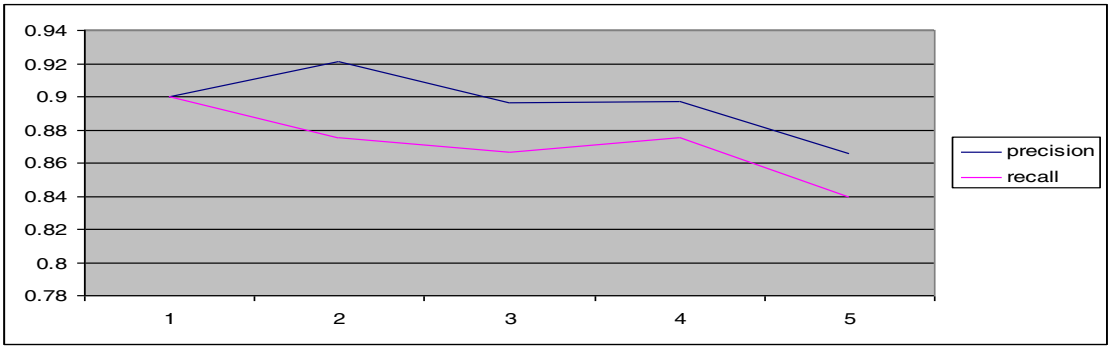

Figure 4.2: Precision-Recall graph

5. Conclusions and possible future extensions:

The paper focuses on developing a natural language interface to the relational database. The inputted query is preprocessed using stop word removal and stemming. The syntactic analysis has done using top-down parser and is represented with first order logic. The system vitally uses the ontology constructs, Parsing rules and FOL logic to extract the requisite information in forming a standard database Query. The system is flexible and can be adapted to any of the Database management systems or a relational database management system. EFFCN algorithm is used to form candidate query for the given natural language query. Using the power of ontology and enhanced parsing mechanisms to filter query up to a refined level where it incorporates needed information as per the user. Compared to which the EFFCN system gives a success rate of 0.84 and high precision of 0.86

The NLIDB system future growth is directed towards improving the success rate by applying concepts of neural networks, machine learning parsing techniques and the use of SQL standard aggregate functions such as average, min and max along with

the operator precedence concepts. The analysis of the system from the perspective of abbreviations and the temporal queries also needs careful interpretation along with the complex restrictions of FOL logic.

6. References:

[1] Mrs. Neelu Nihalani, Dr. Sanjay Silakari and Dr. Mahesh Motwani, "Natural Language Interface for Database: A Brief Review", IJCSI International Journal of Computer Science Issues, vol. 8, no. 2, pp. 600-608, Mar. 2011.

[2] T. Johnson, "Natural Language Computing-The Commercial Applications", The Knowledge Engineering Review, vol. 1, no. 3, pp. 11-23, 1984.

[3] Androutsopoulos, G.D. Ritchie and P. Thanisch, "Natural Language Interface to Databases-An Introduction", Department of Computer Science, University of Edinburgh, King"s Buildings, Mayfield Road, Edinburgh EH9 3JZ, Scotland, U.K. , Mar. 1995.

[4] W.A. Woods, R.M. Kaplan and B.N. Webber, "The Lunar Sciences Natural Language Information System: Final Report", BBN Report 2378, Bolt Beranek and Newman Inc., Cambridge, Massachusetts, 1972.

[5] C.R. Perrault and B.J. Grosz, "Natural Language Interfaces", Exploring Artificial Intelligence, Morgan Kaufmann Publishers Inc., San Mateo, California, 1988, pp. 133-172.

[6] G. Hendrix, E. Sacerdoti, D. Sagalowicz, and J. Slocum, "Developing a Natural Language Interface to Complex Data", ACM Transactions on Database Systems, pp. 105-147, 1978.

[7] W. Woods, "An experimental parsing system for transition network grammars in Natural Language Processing", Algorithmic Press, New York, USA, 1973.

[8] L.R.Harris,"Experience with INTELLECT: Artificial Intelligence Technology Transfer", The AI Magazine, pp. 43-50, 1984.

[9] Faraj A. El-Mouadib, Zakaria S. Zubi, Ahmed A. Almagrous and Irdess S. El-Feghi, "Generic Interactive Natural Language Interface to Databases (GINLIDB)", International Journal of Computers, vol. 3, no. 3, 2009.

[10] "START Natural Language Question Answering System". [Online].Available: http://start.csail.mit.edu/

[11] M. Joshi, R. A. Akerkar, "Algorithms to improve performance of Natural Language Interface", International Journal of Computer Science & Applications, vol. 5, no. 2, pp. 52-68, 2008.

[12] Seymour Knowles and Tanja Mitrovic, "A Natural Language Interface For SQL-Tutor", Nov. 5, 1999.

[13] D.L. Waltz, "An English Language Question Answering System for a Large Relational Database", Communications of the ACM, pp. 526-539, 1978.